



UNDERSTAND AGILE START TO FINISH

* ONE DAY
* ONE TRACK



7:30 Registration

8:30 Initial Remarks

Phil Japikse

9:00 Agile Flavors

Mark Windholtz

Break

9:45 Agile Success Story

Sujit Upadhye

10:15 Engineering Practices

Jim Weirich

Break

11:00 Estimation

Mike Eaton

11:30 2 Things to do tomorrow to be more Agile

Brian Prince

Lunch and Panel Discussion

1:30 Why Agile Fails

Ed Sumerfield and
Chris Nelson

2:00 Enterprise Agility

Phil Japikse

Break

2:45 Budgeting an Agile Project

Matt Van Vleet

3:15 Retrospectives

Joe Obrien

3:45 Business Practices

Mark Windholtz

4:15 Closing Remarks and Raffle

pillar



budget sketch 
GET INTENTIONAL WITH YOUR MONEY



 gaslightsoftware

 ScrumAlliance®
transforming the world of work

Microsoft®

 **ATLASSIAN®**

 **SDS**
Strategic Data Systems

O'REILLY
User group members
SAVE 35% on all titles
Enter Discount Code: DSUG
oreilly.com



MAX
max technical training

 Seapine Software™

 **telerik**
deliver more than expected



Enterprise Agility: Applying Scrum in a Waterfall World

By
Philip Japikse
Phil.japikse@pinnsng.com
MVP, MCSD.Net, MCDBA, CSM, CSP
Principal Consultant
Pinnacle Solutions Group



Who am I?

- ▶ Principal Consultant, Pinnacle Solutions Group
- ▶ Microsoft MVP
- ▶ MCSD, MCDBA, CSM, CSP
- ▶ Enterprise Application Architect
- ▶ Trainer/Mentor/Speaker
- ▶ Lead Director, Cincinnati .NET User's Group
- ▶ President, Agile Conferences, Inc
- ▶ Contributing Author – www.nplus1.org

Can't we all just get along?

- ▶ Teams don't work in isolation
- ▶ Teams must interact with many other groups in the enterprise that
 - Typically are not agile and/or
 - Have no desire/ability to become agile

Making Scrum work

- ▶ Courtesy and Respect
- ▶ Don't just assume they "don't get it"
- ▶ Be "Agile" in interactions

- ▶ Disclaimer: Some of the concepts in the following slides are not traditional Scrum

Release Planning

- ▶ Enterprise projects
 - Usually consist of multiple sprints
 - Require a great deal of coordination between teams
- ▶ Product Backlog must be
 - Complete* (still subject to change)
 - Prioritized (all items, not just top n)
- ▶ Time-box the release
 - Priorities and scope *will* change
 - Estimates will be wrong
- ▶ Involves Product Owner, Architect(s), Security, Infrastructure, QA, etc

Inter-Team Communication

- ▶ Host meetings with representatives from all affected teams on a regular schedule
 - Development team reports:
 - High level progress status
 - Reaffirms architecture
 - Other teams report:
 - Status of infrastructure required for release
 - Any changes to external requirements
- ▶ Meet more often as release gets closer

Swim Lanes


- ▶ Instead of Burn Down Charts
- ▶ “Stolen” from Kanban
- ▶ Tasks/Features move from
 - In Queue
 - In Process
 - Ready for QA
 - Ready for UAT
 - Ready for Release

Refining Requirements

- ▶ A good requirement is one that you can wrap a test around
- ▶ All Backlog items need to be defined well enough that a:
 - Developer can understand and code the intent
 - QA Resource/Tester can validate the code
- ▶ Incomplete items are removed

Wireframes

- ▶ Used to visually layout the User Interface
- ▶ All proposed screens
- ▶ Important to not look “finished”
- ▶ Tools:
 - www.mockupscreens.com
 - www.balsamiq.com



EmpID	Name	Title	Manager
1	Bob	Manager	Sue
2	Sue	VP	
3	Andy	Sales Rep	Bob

OK

User Stories

▶ User Stories

- As an [X] I Want [Y] So That [Z]¹
 - X is a role
 - Y is a feature
 - Z is the benefit

¹<http://dannorth.net/introducing-bdd>

- ▶ As an Account Manager, I want to be able to Edit a Customer's Address so that we can Effectively Communicate with them
- ▶ Includes success criteria

Success Criteria

- ▶ Must be testable
- ▶ Use Given/When/Then syntax
 - Given 2000 customers
 - When selecting one
 - Then the form should open in < 1 second

Context Specification¹

- ▶ When Editing a Customers Address
 - It Should Load in < 1 sec with 2000 customer records
 - It Should allow an Account Manager to edit the address

¹[Behavior Driven Development \(Code Magazine\) – Scott Bellware](#)

Defining “Done”

- ▶ All (Dev, Users, QA, etc) must agree on definition of Done
 - Developer
 - Unit Tests, Documentation, Code Reviews, etc
 - QA
 - Integration Testing, Black Box Testing, etc
 - Users
 - UAT
- ▶ Will be different based on the product
 - NASA vs XBOX

Test/Behavior Driven Development

- ▶ Development needs to be Test Driven
 - QA personnel need to understand what that means
- ▶ Successful T/BDD development teams build confidence in themselves and with others
 - QA shouldn't have to test that
 - Math.Add(2,3) returns 5
 - QA can focus on the bigger picture
 - Making sure the requirements are met
 - Integration Testing

Users

- ▶ Most users/customers don't understand software development
- ▶ Used to waiting months/years to see projects delivered
- ▶ Coaching is required
 - Product Owner is their single Point Of Contact
 - User Testing of Sprints is a new concept

User Testing

- ▶ User Testing is used to validate the state of the software after every sprint.
 - Key Users should be testing the codebase from the previous sprint
 - The Team (via the Product Owner) must fully disclose what they believe to be working and not working
- ▶ Users can enter *potential* defects into the tracking system

QA/Testers

- ▶ Best if QA is part of The Team
 - Corporate Silos can prevent this
- ▶ QA/Testers are used to the waterfall approach
 - Development creates something, throws it over the wall
 - QA tests it, throws it back
 - Ad infinitum
- ▶ Must adopt a different approach to testing

Sprint QA Testing

- ▶ As soon as the Sprint Backlog is determined:
 - Begin creating Test Plans for items in the sprint
 - Create/Update Integration Test Plans for current and previous Sprints
- ▶ When developers believe they are “Done”
 - QA Reviews Unit Tests
 - Validate that they are testing the requirements
- ▶ Bottom line, QA should be *Proactive*, and not *Reactive*

Bug Triage

- ▶ Bug triage meetings happen immediately after the Daily Standup
- ▶ Triage Team
 - Lead QA, Architect/Dev Lead, Product Owner
- ▶ Bugs are marked for either:
 - Sprint Backlog
 - Product Backlog
 - Bug
 - Change Request

Sprint 0

- ▶ Also referred to as the Foundational Sprint
- ▶ Occurs before full Team is formed
 - Product Owner, Application Architect
- ▶ Used for:
 - Configuration (e.g. Build Server, developer Virtuals)
 - Product Backlog creation
 - Acquiring Funding
 - Release/Hardware planning
 - Assembling the Development Team

Verification Sprint

- ▶ Occurs after code “chill”
- ▶ Used for:
 - Security audits
 - Performance/Load/UAT/Integration testing
 - Deployment documentation
- ▶ Team uses this time to work on:
 - Required documentation, improving Unit Tests, etc.
 - NOT refactoring application code

Contact Me

- ▶ phil@skimed.com
- ▶ www.skimed.com/blog
- ▶ www.twitter.com/skimed

Questions?

